Amendments to the Specification

Please replace the paragraph that begins on Page 2, line 10 and carries over to Page 3, line 3 with the following marked-up replacement paragraph:

-- Caching attempts to avoid repeated generation of content by storing content and serving it to subsequent requesters whenever possible. Serving cached content not only reduces the workload on the back-end computing resources, but it also improves response time to the user. Workload balancing improves the performance of a Web site by dynamically adjusting the amount of work sent to each server in a clustered group of servers. Content distribution attempts to pro-actively (statically) publish static content to various locations in the network, for example to cache servers in order to increase the likelihood that requests can be served from cache. Content Distribution Service Providers ("CDSPs") offer a valuable service by providing access to their broad network infrastructure for caching of static content in close proximity to the end user. This, in turn, enables enterprises to scale their operations in a cost-effective manner. Dynamic content distribution (i.e. dynamically moving generated content closer to users) would yield the same scalability benefits. For some applications (e.g. those which provide session management within their presentation logic, and which only access the back-end business logic in batch mode), it may be possible to (statically) deploy the presentation logic at the edge of the network. In these cases, the content distribution process will typically result in reduced response time as well. --

Please replace the paragraph on Page 4, lines 7 - 15 with the following marked-up replacement paragraph:

-- Existing application offload techniques which are known to the present inventors for determining the ~~edgability~~ edgeability of an application component use a two-stage approach. In the first stage, a programmer familiar with the application component describes that component using a single check-off approach that specifies whether the component is edgeable or non-edgeable. This description is then used by a deployer, in the second stage, as input to the final deployment (i.e. edgeability) decision. The deployer may either accept the programmer's edgeability input, or may override the programmer's input, depending (for example) on what the deployer knows about the configuration of the target operating environment. The components are then distributed accordingly. --

Please replace the paragraph that begins on Page 7, line 16 and carries over to Page 8, line 1 with the following marked-up replacement paragraph:

-- Figure 3 (comprising Figures 3A and 3B) illustrates ~~illustrate~~ use of several example program characteristics, and Figure 4 (comprising Figures 4A and 4B) illustrates ~~illustrate~~ use of several operating environment characteristics, for determining edgeability of program components, according to preferred embodiments of the present invention; --